Preliminaries:

Before you begin, ensure you have done the following (instructions are contained within the slides):

- Loaded your script from yesterday, **problems.R** into RStudio.
- Loaded your session, **problems.RData** into RStudio.
- Set the working directory as the r_course folder

Your dataset should contain the entire dataframe for colon_cancer_data_set.txt, and two subsets, one containing gene expression for the tumour (affected) samples and other containing data for the normal (unaffected) samples.

Note, you can use the `ls()` function to list the variables contained within your R session. Alternatively (or simultaneously), you can look at the **environment window** (top right) in RStudio, which should also list the contents of your session.

Q1. Generate two random variables containing 100 data points using the `rnorm()` function. Note, to replicate these exact variables, you will have to set the seed to any number (`set.seed()`).

Using these two random variables:

1.1 Perform a t-test (`t.test()`) to test the differences in means. Extract the p-value from the `t.test()` results.

1.2 Perform a Wilcoxon Test (`wilcox.test()`) to test the difference in means. Extract the p-value from the `wilcox.test()` results.

1.3 Using the `cor()` function compute the Pearson's r and the Spearman's rho for the two random variables.

1.4 Install and load the package perm. Using the code below, run permutation two sample test on the random variables. Extract the p-value from the permutation test. Note, this

approach to permutation only works on small small sample sizes. With large sample sizes, the Wilcoxon test and the T-test are quite robust.

```
perm_test <- permTS(var1,var2,
              alternative="two.sided",
              method="exact.ce",
              control=permControl(tsmethod="abs"))
```

Q2. In this problem, we will perform statistical tests comparing the expression of three genes, guanylin, pyrroline reductase and apolipoprotein A in normal and tumour cells.

2.1 Construct a FOR loop to print (`print()`) the normal and tumour gene expression to the screen. (**Hint**: Both the `unaffected_genes` and `affected_genes` have the same dimensions and the same order of genes – by iterating over one, you can access the corresponding gene in the other).

2.2 Replace the print statements with variables holding the following:

- The p-value from a two-tailed paired t-test.
- The p-value from a paired Wilcoxon test.
- The Pearson's r.
- The Spearman's rho.

2.3 Modify the FOR loop to combine the values from the above test into a vector and append the vector to a holder (**Hint:** `rbind()`).

2.4 Look at the holder data. Check what kind of data structure holder is (Hint: `class()`). Convert holder to a dataframe (**Hint**:

`as.data.frame()`) Change the column titles to 't_test', 'w_test', 'pear_r', 'spear_rho'. Change the row names to 'guanylin', 'pyrroline and 'apolipoprotein'.

The aim of this problem is to achieve a data frame of the form:

	t_test	w_test	pear_r	spear_r
guanylin	p-values	p-values	r	rho
pyrroline	p-values	p-values	r	rho
apolipoprotein	p-values	p-values	r	rho

2.5 Look at the values returned by the tests. All of the distributions used in this analysis have departures from normality. Which p and which r would you trust? Are non-parametric p-values generally more conservative?

Q3. Once completed:

3.1 Save your script.

3.2 Save your session.

Note! Please ensure to save your work as we will use this data set continuously throughout the course.